Quantstamp Contract Security Certificate

December 18th 2019 — Quantstamp Verified

POA Network - Extended

This smart contract audit was prepared by Quantstamp, the protocol for securing smart contracts.

Executive Summary

Type	Integration
Auditors	Leonardo Passos, Senior Research Engineer Ed Zulkoski, Senior Security Engineer Martin Derka, Senior Research Engineer
Timeline	2019-11-19 through 2019-12-16
EVM	Constantinople
Languages	Solidity
Methods	This audit extends our initial audit released on Nov. 18th, 2019, which only focused on commit 385faac8. The scope of this audit now includes all non-test-related Solidity files.
Specification	None

Source Code

Repository	Commit
poanetwork	c2c2c4cc2b530

Changelog

- 2019-11-29 - Initial report (commit 385faac8a88c6)
- 2019-12-02 - Diff audit (commit c2c2c4cc2b530)

Overall Assessment

- There is a lot of assembly code that could be replaced by simple contract calls. The complexity of assembly programming poses risk that in our opinion could have been easily prevented;
- Many functions rely on access modifiers that add considerable centralization power, requiring user trust; on the other hand, some functions we had expected to have privileged access modifiers did not have any, making the protocol vulnerable;
- High risk vulnerabilities have been identified and require prompt attention from POA developers;
- External documentation is too high-level for a proper understanding of all the details in place. Moreover, the code is poorly documented, which imposed the audit team with great challenge in understanding it, specially under a tight deadline (~ 2 weeks).

Total Issues	**15**	(5 Fixed)
High Risk Issues	**3**	(3 Fixed)
Medium Risk Issues	0	(0 Fixed)
Low Risk Issues	**4**	(1 Fixed)
Informational Risk Issues	**5**	(0 Fixed)
Undetermined Risk Issues	**3**	(1 Fixed)

High Risk	The issue puts a large number of users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
Medium Risk	The issue puts a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or is reasonably likely to lead to moderate financial impact.
Low Risk	The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low-impact in view of the client's business circumstances.
Informational	The issue does not post an immediate risk, but is relevant to security best practices or Defence in Depth.
Undetermined	The impact of the issue is uncertain.
Unfixed	Acknowledged the existence of the risk, and decided to accept it without engaging in special efforts to control it.
Acknowledged	the issue remains in the code but is a result of an intentional business or design decision. As such, it is supposed to be addressed outside the programmatic means, such as: 1) comments, documentation, README, FAQ; 2) business processes; 3) analyses showing that the issue shall have no negative consequences in practice (e.g., gas analysis, deployment settings).
Fixed	Adjusted program implementation, requirements or constraints to eliminate the risk.

Summary of Findings

ID	Description	Severity	Status
QSP-1	Denial-of-Service (DoS)	High	Fixed
QSP-2	Fee Distribution Can Be Stolen	High	Fixed
QSP-3	Anyone Can Set Fees in the Home/Foreign Side of a Bridge	High	Fixed
QSP-4	Sending Balance to Unknown Contract	Low	Fixed
QSP-5	`distributeFeeProportionally` Can be Manipulated	Low	Acknowledged
QSP-6	Inconsistent Logic in Transfer Limit Set Functions	Low	Acknowledged
QSP-7	Bridges May not be Singletons for a Given Token	Low	Acknowledged
QSP-8	Centralization of Power	Informational	Acknowledged
QSP-9	Initialize Functions are Publicly Accessible	Informational	Acknowledged
QSP-10	Length of `_bytes` Parameter is Unchecked	Informational	Acknowledged
QSP-11	Clone-and-Own	Informational	Acknowledged
QSP-12	Allowance Double-Spend Exploit	Informational	Acknowledged
QSP-13	Incorrect Versioning	Undetermined	Fixed
QSP-14	Iterating over List of Validators May Fail	Undetermined	Acknowledged
QSP-15	Gas Usage / `for` Loop Concerns	Undetermined	Acknowledged

Quantstamp Audit Breakdown

Quantstamp's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices.

Toolset

The notes below outline the setup and steps performed in the process of this audit.

Setup

Tool Setup:

- Truffle
- Ganache
- SolidityCoverage
- Mythril
- Slither

Steps taken to run the tools:

1. Installed Truffle: `npm install -g truffle`
2. Installed Ganache: `npm install -g ganache-cli`
3. Installed the solidity-coverage tool (within the project's root directory): `npm install --save-dev solidity-coverage`
4. Ran the coverage tool from the project's root directory: `./node_modules/.bin/solidity-coverage`
5. Installed the Mythril tool from Pypi: `pip3 install mythril`
6. Ran the Mythril tool on each contract: `myth -x path/to/contract`
7. Installed the Slither tool: `pip install slither-analyzer`
8. Run Slither from the project directory `slither .`

Assessment

Findings

QSP-1 Denial-of-Service (DoS)

Severity: *High Risk*

Status: Fixed

File(s) affected: `contracts/upgradeable_contracts/erc20_to_native/ForeignBridgeErcToNative.sol`

Description: A Denial-of-Service (DoS) attack is a situation which an attacker renders a smart contract unusable.

Exploit Scenario: On L92 (commit `385faac8a88c6`) `uint256 curBalance = erc20token().balanceOf(address(this))`, one gets the current *SAI* balance of this contract, and then L100 checks that the full balance has been transferred to the *DAI* contract. However, if the contract has a current DAI balance (e.g., simply from ERC20 DAI transfers from any user), this check will fail, as the DAI balance will be higher than what gets converted.

Recommendation: Instead of checking strict balance equality, check if the resulting balance is at least the same as `curBalance`.

QSP-2 Fee Distribution Can Be Stolen

Severity: *High Risk*

Status: Fixed

File(s) affected: `contracts/upgradeable_contracts/erc20_to_erc20/FeeManagerErcToErcPOSDAO.sol`

Description: When a rewardable bridge is initialized (e.g., `HomeBridgeNativeToErc`), it takes a `_feeManager` parameter - say `FeeManagerErcToErcPOSDAO` contract address. The latter, however, allows *ANYONE* to override the block reward contract by invoking `setBlockRewardContract`.

Exploit Scenario: (1) `Mallory` (attacker) calls `setBlockRewardContract` on `FeeManagerErcToErcPOSDAO`, setting its own block reward contract. Mallory's contract works in a way that whenever fees are to be distributed, it keeps all fees to itself. Additionally, Mallory's contract contains a function `blockRewardContractId`, which returns `0x0d35a7ca` (`bytes4(keccak256("blockReward"))`).
(2) When `Mallory` sets his contract, it triggers the call to `BlockRewardBridge:_setBlockRewardContract`. In it, since Mallory's `blockRewardContractId` contract function returns `0x0d35a7ca`, the following code executes:

```
if (_blockReward.call(BLOCK_REWARD_CONTRACT_ID)) {
  isBlockRewardContract =
      IBlockReward(_blockReward).blockRewardContractId() == bytes4(keccak256("blockReward"));
}
```

followed by

```
require(isBlockRewardContract);
addressStorage[BLOCK_REWARD_CONTRACT] =  blockReward
```

Thus, Mallory is able to pass the if-condition and set its own block reward contract. Consequently, when fees are to be distributed, Mallory's contract essentially keeps all the fees, or distribute them according to his own will.

Recommendation: Place a modifier in `setBlockRewardContract`, making it a privileged operation.

> Reply from POA: The bridge contract operates with methods of the fee manager contract in similar way how a library's methods are invoked - through `delegatecall()` and `callcode()`. It means that the fee manager's methods are executed in the context of the bridge contract and operate with the bridge contract's storage. If the block reward contract address is set as part of the `initialize()` invocation it is stored in the storage of the bridge contract. If later someone calls `setBlockRewardContract()` of the fee manager (in the context of the fee manager contract) it will not affect the storage in the bridge contract and the block reward contract address will remain the same and the fees distribution operation will operate with the contract accessed from the bridge contract storage.

QSP-3 Anyone Can Set Fees in the Home/Foreign Side of a Bridge

Severity: *High Risk*

Status: Fixed

File(s) affected: `contracts/upgradeable_contracts/BaseFeeManager.sol`

Description: `setHomeFee` and `setForeignFee` functions do not have any access modifier; consequently, *ANYONE* can set the fees in the home/foreign side of the bridge.

Recommendation: Place a modifier in `setHomeFee` and `setForeignFee` functions, making them privileged.

> Reply from POA: `setHomeFee()` and `setForeignFee()` are intended to be called in the context of the bridge contract by `delegatecall()` in order to configure the fee percentage. If these methods are called in the context of the fee manager they will not impact on the fees calculation.

QSP-4 Sending Balance to Unknown Contract

Severity: *Low Risk*

Status: Fixed

File(s) affected: `contracts/upgradeable_contracts/erc20_to_native/ForeignBridgeErcToNative.sol`

Description: In `migrateToMCD` (commit `c2c2c4cc2b530`), the contract approves and sends the entire balance to the migration contract; the latter is received as an address parameter: `_migrationContract`. There is no guarantee that the informed migration contract will migrate SAI to DAI. It may only accept the SAI balance and do nothing. Extreme caution should be taken to provide the address of the official migration contract.

Recommendation: Wait for the official address release and hardcode the address. Alternatively, make this address configurable within the contract (in case of updates).

QSP-5 `distributeFeeProportionally` Can be Manipulated

Severity: *Low Risk*

Status: Acknowledged

File(s) affected: `contracts/upgradeable_contracts/ValidatorsFeeManager.sol`

Related Issue(s): SWC-120

Description: `distributeFeeProportionally` relies on a call to `random(numOfValidators)`, whose implementation is given by `uint256(blockhash(block.number.sub(1))) % _count`, where `_count` is an `uint256` parameter. If a validator also happens to be a miner or colludes with one or more, that specific validator can be given the `diff` value (if available) multiple times, making the distribution uneven.

Recommendation: Everything in the blockchain is visible and publicly available; thus, any reliable randomization must use an outside random data source (e.g., an oracle).

> Reply from POA: We know about the possibility to have uneven fee distribution. But it is a trade off between increasing the complexity and possible negative impact. Consider the situation when the bridge is operated by 3 validators. The maximum difference in the fee distribution between them for one transfer is 2 wei. For 5 validators it will be 4 wei. At the same time the minimal value that can be transferred through the bridge usually are greater than 0.001 ether. So, in order to have any significant impact to expose the fact that the fees are not distributed proportionally the validator needs to have opportunity to impact on mining of billions of blocks.

QSP-6 Inconsistent Logic in Transfer Limit Set Functions

Severity: *Low Risk*

Status: Acknowledged

File(s) affected: `contracts/upgradeable_contracts/BasicTokenBridge.sol`

Description: When initializing a bridge, constructors invoke the `_initialize` function, which among others, check certain constraints concerning transaction limits, namely:

- `minPerTx > 0`;
- `maxPerTx > minPerTx`;
- `dailyLimit > maxPerTx` (thus, `daily limit cannot be zero`);
- `homeMaxPerTx` < `homeDailyLimit` (thus, execution daily limit cannot be zero).

However, set-like functions in `BasicTokenBridge.sol` do not check the same constraints, namely:

- `setMinPerTx` does not check if given parameter is greater than zero;
- `setMaxPerTx` allows setting a value lower or equal to `minPerTx()`;
- `setDailyLimit` allows setting a zero limit;
- `setExecutionDailyLimit` allows setting a zero limit.

Recommendation: Enhance requirement conditions to guarantee set-like functions match initialization logic.

> Reply from POA: A pull request (PR) has been created but it has not yet been merged. PR is available at https://github.com/poanetwork/tokenbridge-contracts/issues/322

QSP-7 Bridges May not be Singletons for a Given Token

Severity: *Low Risk*

Status: Acknowledged

File(s) affected: `contracts/upgradeable_contracts/erc20_to_erc20/*`,

Description: Currently, there is no mechanism to control transfer limits for a given token X. If one wants to bypass any given limit, all they have to do is spin-up a new bridge for that particular token.

Recommendation: Enhance code documentation to better orient users.

> Reply from POA: There is no way to prevent others to deploy their own bridges to operate with a token. If someone deployed another bridge validators most probably will not confirm the relay operations of that bridge at all. Validator's oracles need to be configured explicitly to watch for events from specific bridge/token contracts. It is assumed that each bridge has its own set of validators.

QSP-8 Centralization of Power

Severity: *Informational*

Status: Acknowledged

File(s) affected: `contracts/upgradeable_contracts/erc20_to_native/ForeignBridgeErcToNative.sol`

Description: Smart contracts will often have `owner` variables to designate the person with special privileges to make modifications to the smart contract. However, this centralization of power needs to be made clear to the users, especially depending on the level of privilege the contract allows to the owner.

Exploit Scenario: In commit `c2c2c4cc2b530`, the method `migrateToMCD` is protected by `onlyOwner`. Users cannot trigger the migration on their own, and there is no guarantee that the owner will call this method.

Recommendation: Related to `Sending balance to unknown contract` issue, once the migration contract address is known, there is no need to receive it as a parameter. In the latter case, if the migration address is hardcoded or configurable, the `migrateToMCD` function does not need the `onlyOwner` modifier and anyone could invoke `migrateToMCD`, and thus trigger the migration.

> Reply from POA: The modifier will be removed after merging pull request 319 - https://github.com/poanetwork/tokenbridge-contracts/pull/319.

QSP-9 Initialize Functions are Publicly Accessible

Severity: *Informational*

Status: Acknowledged

Description: `initialize` functions are marked as `external` with no access control modifier. Additionally, some of the these functions receive other contracts as parameters (e.g., `_validatorContract`, `_bridgeContract`, etc), which could allow attackers to initialize already deployed contracts and hook their own logic.

Recommendation: Make sure your deployment scripts (not in the scope of this audit) wire contracts correctly, and that all contracts deployed by POA are indeed initialized as expected, i.e., after the constructor executes, initialization is *NOT* performed by any unknown/unexpected party. Moreover, make sure that deployment scripts assert the initialization led to the expected state.

> Reply from POA: An issue has been created to address this finding. See https://github.com/poanetwork/tokenbridge-contracts/issues/323.

QSP-10 Length of `_bytes` Parameter is Unchecked

Severity: *Informational*

Status: Acknowledged

File(s) affected: `contracts/libraries/Bytes.sol`

Description: The `_bytes` parameter in both `bytesToBytes32` and `bytesToAddress` functions may not have the expected length (32, and 20, respectively). Otherwise, depending on the input, the result may be padded or truncated.

Recommendation: In `bytesToBytes32`, one should check that the length of the `_bytes` parameter is 32. Likewise, on `bytesToAddress`, one should check that the length of the input parameter is 20.

> Reply from POA: `bytesToBytes32` is used in three places: (i) `contracts/upgradeable_contracts/amb_erc677_to_erc677/BasicAMBErc677ToErc677.sol`, method `nonce`; (ii) `contracts/upgradeable_contracts/arbitrary_message/MessageProcessor.sol`, method `failedMessageDataHash`; (iii) `contracts/upgradeable_contracts/arbitrary_message/MessageProcessor.sol`, method `transactionHash`. These three methods are getters for the values that are stored by setters: `setNonce`, `setFailedMessageDataHash` and `setTransactionHash`. All three setters explicitly operates with data of 32 bytes long. So, there is no case when set of bytes with the size different from 32 bytes will be used in `bytesToBytes32` method - additional check for the size will consume gas without adding security. bytesToAddress is used only in `contracts/upgradeable_contracts/BaseERC677Bridge.sol`, method `chooseReceiver`. You will see in the code that checks if the length of the bytes set is 20. So, no additional check is required.

Final recommendation: We still argue that this is an issue. Since `Bytes` is a reusable library, its use is not restricted to POA. So, while it may work as is for POA, it may not work for others, or it could cause future bugs as POA evolves, e.g. say a developer forgets to perform the checks needed to be done prior to calling the functions in `Bytes`, or operates with inputs whose length differ from those that are expected, or simply is not aware of length requirements. For the time being, POA agreed to enhance the documentation of both `bytesToBytes32` and `bytesToAddress` to document what the expected output should look like upon having input of different lengths.

QSP-11 Clone-and-Own

Severity: *Informational*

Status: Acknowledged

File(s) affected: `upgradeability/*`

Description: The clone-and-own approach involves copying and adjusting open source code at one's own discretion. From the development perspective, it is initially beneficial as it reduces the amount of effort. However, from the security perspective, it involves some risks as the code may not follow the best practices, may contain a security vulnerability, or may include intentionally or unintentionally modified upstream libraries. Rather than the clone-and-own approach, a good industry practice is to use the Truffle framework for managing library dependencies. This eliminates the clone-and-own risks yet allows for following best practices, such as, using libraries.

Recommendation: Since upgradeability clones most of the behavior in https://github.com/OpenZeppelin/openzeppelin-labs/tree/master/upgradeability_using_eternal_storage, we recommend using the latter as a dependency, rather than cloning its code.

> Reply from POA: An issue has been created to address this finding. See https://github.com/poanetwork/tokenbridge-contracts/issues/324.

QSP-12 Allowance Double-Spend Exploit

Severity: *Informational*

Status: Acknowledged

File(s) affected: `contracts/ERC677BridgeToken.sol`, `contracts/ERC677BridgeTokenRewardable.sol`

Description: As it presently is constructed, the contract is vulnerable to the allowance double-spend exploit, as with other ERC20 tokens. An example of an exploit goes as follows:

1. Alice allows Bob to transfer `N` amount of Alice's tokens (`N>0`) by calling the `approve()` method on `Token` smart contract (passing Bob's address and `N` as method arguments)
2. After some time, Alice decides to change from `N` to `M` (`M>0`) the number of Alice's tokens Bob is allowed to transfer, so she calls the `approve()` method again, this time passing Bob's address and `M` as method arguments
3. Bob notices Alice's second transaction before it was mined and quickly sends another transaction that calls the `transferFrom()` method to transfer `N` Alice's tokens somewhere
4. If Bob's transaction will be executed before Alice's transaction, then Bob will successfully transfer `N` Alice's tokens and will gain an ability to transfer another `M` tokens
5. Before Alice notices any irregularities, Bob calls `transferFrom()` method again, this time to transfer `M` Alice's tokens. The exploit (as described above) is mitigated through use of functions that increase/decrease the allowance relative to its current value, such as `increaseAllowance` and `decreaseAllowance`.

Pending community agreement on an ERC standard that would protect against this exploit, we recommend that developers of applications dependent on `approve()` / `transferFrom()` should keep in mind that they have to set allowance to 0 first and verify if it was used before setting the new value. Teams who decide to wait for such a standard should make these recommendations to app developers who work with their token contract.

Recommendation: There is no fix for this issue; adding `increase/decrease` allowance functions mitigates the issue.

> Reply from POA: An issue has been created to address this finding. See https://github.com/poanetwork/tokenbridge-contracts/issues/325.

QSP-13 Incorrect Versioning

Severity: *Undetermined*

Status: Fixed

File(s) affected: `contracts/upgradeable_contracts/VersionableBridge.sol`, `contracts/upgradeable_contracts/BaseBridgeValidators.sol`

Description: `VersionableBridge:getBridgeInterfacesVersion()` is not consistent with `BaseBridgeValidators:getBridgeValidatorsInterfacesVersion()`, which in turn does not seem consistent with the latest version of the project.

Recommendation: Update version values to match the latest project version.

> Reply from POA: VersionableBridge:getBridgeInterfacesVersion() and BaseBridgeValidators:getBridgeValidatorsInterfacesVersion() do not match, and this is intentional, as they control different matters.

QSP-14 Iterating over List of Validators May Fail

Severity: *Undetermined*

Status: Acknowledged

File(s) affected: `contracts/upgradeable_contracts/BaseBridgeValidators.sol`, `contracts/upgradeable_contracts/ValidatorsFeeManager.sol`

Description: If the list of validators grows excessively, iterating over its elements may be prohibitive due to gas cost. Thus, the following functions may fail:

- `BaseBridgeValidators:_removeValidator`
- `ValidatorsFeeManager:distributeFeeProportionally`

Recommendation: Partition the set of validators such that each partition contains a small number of validators. Thus, `BaseBridgeValidators:_addValidator` would add a validator to the first partition that has an empty slot (consequently, one must control the size of each partition). Likewise, removal would require keeping an inverted list to map a given validator to its partition. Then, removal would iterate that partition only. In the case of fee distribution, only a given partition is rewarded; the first time, all the validators in the first partition get the reward, the second time validators in the second partition get the reward, and so on and so forth, until the last partition is rewarded. The cycle then re-starts from the first partition. An alternative solution is to let validators claim their rewards.

> Reply from POA: An issue has been created to address this finding. See https://github.com/poanetwork/tokenbridge-contracts/issues/326.

QSP-15 Gas Usage / `for` Loop Concerns

Severity: *Undetermined*

Status: Acknowledged

File(s) affected: `contracts/libraries/Message.sol`

Description: Gas usage is a main concern for smart contract developers and users, since high gas costs may prevent users from wanting to use the smart contract. Even worse, some gas usage issues may prevent the contract from providing services entirely. For example, if a `for` loop requires too much gas to exit, then it may prevent the contract from functioning correctly entirely. It is best to break such loops into individual functions as possible.

Recommendation: Keep number of required signatures low; otherwise `BasicForeignAMB:executeSignatures` may consistently fail.

> Reply from POA: An issue has been created to address this finding. See https://github.com/poanetwork/tokenbridge-contracts/issues/327.

Automated Analyses

Mythril

We have analyzed each vulnerability reported by Mythril, filtering out false-positives; those remaining after the filtering (true vulnerabilities) have been included as part of this report.

Slither

We have analyzed each vulnerability reported by Slither, filtering out false-positives; those remaining after the filtering (true vulnerabilities) have been included as part of this report.

Adherence to Best Practices

- Old solidity version being used. We recommend migrating to a newer version.
- `require` statements don't have a string message; provide one.
- Functions `setErc677Token` allows resetting the token. Setting should occur at most one time - assert it!
- Functions `setErc20Token` allows resetting the token. Setting should occur at most one time - assert it!
- Many parameters in POA are resettable after deployment, and may only be changed by privileged users -- a representative example is `BasicBridge.sol`. While such strategy allows flexibility in tuning the protocol post-deployment, it does introduce power-centralization. Make sure you clearly communicate when parameters are updated and the rationale for doing so.
- Avoid using assembly code whenever possible. While this could potentially save gas, it does introduce further complexity and does make (human) reasoning harder. At times, it just seems completely unnecessary. For instance, in `Claimable.sol`, what is the purpose of `safeTransfer()`, as opposed to simply invoking `transfer()` and checking the result? Please review your code and favor eliminating assembly with Solidity code, unless there is a significant reason not to do so.
- On L102 of `BasicAMBErc677ToErc677.sol`, the return value of `token.transferFrom(_from, to, _value)` should be checked to be true.
- On L9,20 of `HomeAMBErc677ToErc677.sol`, the return value `IBurnableMintableERC677Token(erc677token()).mint(_recipient,_value)` is not checked.
- On L26 of `ERC20Bridge.sol`, the return value of `erc20token().transferFrom()` is not checked.
- `RewardableValidators.sol`: the contract definition should indicate that it implements the `IRewardableValidators` interface.
- In `ForeignBridgeErcToNative`, `isHDTokenBalanceAboveMinBalance` is comparing whether the SAI balance of the current contract is equal to the value returned by `minHDTokenBalance`, but the latter is set to be in Ether (see line 107). Strictly, it works, but the intention is unclear. Why would you compare Eth to SAI in absolute values? If that is what developers intended, it would have sufficed to set `uintStorage[MIN_HDTOKEN_BALANCE]` to `10000000000000000000`.
- In `ForeignBridgeErcToNative`, the assignment on line 107 relies on a magic constant, but no documentation is provided to justify rationale. Please add proper documentation to enhance understanding.

Test Results

Test Suite Results

```
  Contract: ForeignAMBErc677ToErc677
    initialize
       ✓ should initialize (926ms)
       ✓ only owner can set bridge contract (251ms)
       ✓ only owner can set mediator contract (227ms)
       ✓ only owner can set request Gas Limit (212ms)
    set limits
       ✓ setMaxPerTx allows to set only to owner and cannot be more than daily limit (112ms)
       ✓ setMinPerTx allows to set only to owner and cannot be more than daily limit and should be less than
maxPerTx (114ms)
       ✓ setDailyLimit allow to set by owner and should be greater than maxPerTx or zero (218ms)
       ✓ setExecutionMaxPerTx allows to set only to owner and cannot be more than daily limit (112ms)
       ✓ setExecutionDailyLimit allow to set by owner and should be greater than maxPerTx or zero (205ms)
    getBridgeMode
```

```
      ✓ should return arbitrary message bridging mode and interface
    fixAssetsAboveLimits
      ✓ Should revert if value to unlock is bigger than max per transaction (41ms)
      ✓ Should allow to partially reduce outOfLimitAmount and not emit amb event (178ms)
      ✓ Should allow to partially reduce outOfLimitAmount and emit amb event (216ms)
      ✓ Should revert if try to unlock more than available (191ms)
      ✓ Should not be allow to be called by an already fixed txHash (146ms)
      ✓ Should fail if txHash didnt increase out of limit amount (42ms)
      ✓ Should fail if not called by proxyOwner (89ms)
    relayTokens
      ✓ should allow to bridge tokens using approve and transferFrom (244ms)
      ✓ should allow user to specify a itself as receiver (287ms)
      ✓ should allow to specify a different receiver (263ms)
      ✓ should allow to specify a different receiver without specifying sender (251ms)
      ✓ should allow to complete a transfer approved by other user (282ms)
      ✓ should fail if user did not approve the transfer (92ms)
      ✓ should fail if value is not within limits (128ms)
      ✓ should prevent emitting the event twice when ERC677 used by relayTokens and ERC677 is owned by token
manager (398ms)
      ✓ should prevent emitting the event twice when ERC677 used by relayTokens and ERC677 is not owned by token
manager (331ms)
    requestFailedMessageFix
      ✓ should allow to request a failed message fix (165ms)
      ✓ should be a failed transaction (98ms)
      ✓ should be the receiver of the failed transaction (99ms)
      ✓ message sender should be mediator from other side (112ms)
      ✓ should allow to request a fix multiple times (295ms)
  fixFailedMessage
      ✓ should fix burnt/locked tokens (282ms)
      ✓ should be called by bridge
      ✓ message sender should be mediator from other side (265ms)
  #claimTokens
      ✓ should be able to claim tokens (314ms)
  onTokenTransfer
      ✓ should emit UserRequestForAffirmation in AMB bridge (338ms)
      ✓ should be able to specify a different receiver (558ms)
  handleBridgedTokens
      ✓ should transfer locked tokens on message from amb (363ms)
      ✓ should transfer locked tokens on message from amb with decimal shift of two (550ms)
      ✓ should emit AmountLimitExceeded and not transfer tokens when out of execution limits (362ms)

Contract: HomeAMBErc677ToErc677
  initialize
      ✓ should initialize (594ms)
      ✓ only owner can set bridge contract (190ms)
      ✓ only owner can set mediator contract (183ms)
      ✓ only owner can set request Gas Limit (164ms)
  set limits
      ✓ setMaxPerTx allows to set only to owner and cannot be more than daily limit (104ms)
      ✓ setMinPerTx allows to set only to owner and cannot be more than daily limit and should be less than
maxPerTx (103ms)
      ✓ setDailyLimit allow to set by owner and should be greater than maxPerTx or zero (192ms)
      ✓ setExecutionMaxPerTx allows to set only to owner and cannot be more than daily limit (106ms)
      ✓ setExecutionDailyLimit allow to set by owner and should be greater than maxPerTx or zero (190ms)
    getBridgeMode
      ✓ should return arbitrary message bridging mode and interface
    fixAssetsAboveLimits
      ✓ Should revert if value to unlock is bigger than max per transaction
      ✓ Should allow to partially reduce outOfLimitAmount and not emit amb event (158ms)
      ✓ Should allow to partially reduce outOfLimitAmount and emit amb event (178ms)
      ✓ Should revert if try to unlock more than available (195ms)
      ✓ Should not be allow to be called by an already fixed txHash (136ms)
      ✓ Should fail if txHash didnt increase out of limit amount (39ms)
      ✓ Should fail if not called by proxyOwner (91ms)
    relayTokens
      ✓ should allow to bridge tokens using approve and transferFrom (240ms)
      ✓ should allow user to specify a itself as receiver (237ms)
      ✓ should allow to specify a different receiver (238ms)
      ✓ should allow to specify a different receiver without specifying sender (255ms)
      ✓ should allow to complete a transfer approved by other user (302ms)
      ✓ should fail if user did not approve the transfer (91ms)
      ✓ should fail if value is not within limits (165ms)
      ✓ should prevent emitting the event twice when ERC677 used by relayTokens and ERC677 is owned by token
manager (424ms)
      ✓ should prevent emitting the event twice when ERC677 used by relayTokens and ERC677 is not owned by token
manager (326ms)
    requestFailedMessageFix
      ✓ should allow to request a failed message fix (164ms)
      ✓ should be a failed transaction (87ms)
      ✓ should be the receiver of the failed transaction (80ms)
      ✓ message sender should be mediator from other side (86ms)
      ✓ should allow to request a fix multiple times (269ms)
  fixFailedMessage
      ✓ should fix burnt/locked tokens (293ms)
      ✓ should be called by bridge
      ✓ message sender should be mediator from other side (284ms)
  #claimTokens
      ✓ should be able to claim tokens (289ms)
  onTokenTransfer
      ✓ should emit UserRequestForSignature in AMB bridge and burn transferred tokens (375ms)
      ✓ should be able to specify a different receiver (436ms)
  handleBridgedTokens
      ✓ should mint tokens on message from amb (445ms)
      ✓ should mint tokens on message from amb with decimal shift of two (577ms)
      ✓ should emit AmountLimitExceeded and not mint tokens when out of execution limits (419ms)

Contract: ForeignAMB
  getBridgeMode
      ✓ should return arbitrary message bridging mode (49ms)
  initialize
      ✓ sets variables (255ms)
      ✓ should fail with invalid arguments (217ms)
      ✓ can update variables (332ms)
```

```
  upgradeable
     ✓ can be upgraded (215ms)
     ✓ can be deployed via upgradeToAndCall (130ms)
     ✓ can transfer ownership (193ms)
  requireToPassMessage
     ✓ call requireToPassMessage(address, bytes, uint256)
  executeSignatures
     ✓ should succeed on Subsidized mode (183ms)
     ✓ test with 3 signatures required (440ms)
     ✓ should not allow to double execute signatures (200ms)
     ✓ should allow non-authorities to execute signatures (109ms)
     ✓ status of RelayedMessage should be false on contract failed call (265ms)
     ✓ status of RelayedMessage should be false on contract out of gas call (267ms)

Contract: HomeAMB
  getBridgeMode
     ✓ should return arbitrary message bridging mode and interface (67ms)
  initialize
     ✓ sets variables (269ms)
     ✓ should fail with invalid arguments (219ms)
     ✓ can update variables (335ms)
  upgradeable
     ✓ can be upgraded (223ms)
     ✓ can be deployed via upgradeToAndCall (165ms)
     ✓ can transfer ownership (261ms)
  requireToPassMessage
     ✓ call requireToPassMessage(address, bytes, uint256)
     ✓ call requireToPassMessage(address, bytes, uint256) should fail (209ms)
  executeAffirmation
     ✓ should succeed on Subsidized mode (171ms)
     ✓ test with 3 signatures required (478ms)
     ✓ should not allow to double execute (175ms)
     ✓ should not allow non-authorities to execute affirmation (158ms)
     ✓ status of AffirmationCompleted should be false on contract failed call (240ms)
     ✓ status of AffirmationCompleted should be false on contract out of gas call (222ms)
  submitSignature
     ✓ allows a validator to submit a signature (141ms)
     ✓ test with 3 signatures required (433ms)
     ✓ should not allow to double submit (258ms)
     ✓ should not allow non-authorities to submit signatures (180ms)

Contract: ForeignBridge_ERC20_to_ERC20
  #initialize
     ✓ should initialize (617ms)
  #executeSignatures
     ✓ should allow to executeSignatures (123ms)
     ✓ should allow second withdrawal with different transactionHash but same recipient and value (208ms)
     ✓ should not allow second withdraw (replay attack) with same transactionHash but different recipient
(166ms)
     ✓ should not allow withdraw over home max tx limit (87ms)
     ✓ should not allow withdraw over daily home limit (225ms)
  #withdraw with 2 minimum signatures
     ✓ withdraw should fail if not enough signatures are provided (131ms)
     ✓ withdraw should fail if duplicate signature is provided (76ms)
     ✓ works with 5 validators and 3 required signatures (318ms)
  #upgradeable
     ✓ can be upgraded (518ms)
     ✓ can be deployed via upgradeToAndCall (157ms)
  #claimTokens
     ✓ can send erc20 (423ms)
  #ForeignBridgeErc677ToErc677_onTokenTransfer
     ✓ should emit correct events on initialize (121ms)
     ✓ can only be called from token contract (312ms)
     ✓ should not allow to transfer more than maxPerTx limit (369ms)
     ✓ should only let to transfer within daily limit (492ms)
     ✓ should not let to transfer less than minPerTx (432ms)
     ✓ should be able to specify a different receiver (367ms)
  #decimalShift
     ✓ Home to Foreign: withdraw with 1 signature with a decimalShift of 2 (277ms)
     ✓ Home to Foreign : withdraw works with 5 validators and 3 required signatures with a decimalShift of 2
(513ms)
     ✓ Foreign to Home: no impact in UserRequestForAffirmation event signal for bridges oracles with a
decimalShift of 2. (535ms)
  #relayTokens
     ✓ should allow to bridge tokens using approve tranferFrom (239ms)
     ✓ should allow to call relayTokens without specifying the sender (196ms)
     ✓ should not be able to transfer more than limit (240ms)
     ✓ should allow only sender to specify a different receiver (352ms)

Contract: HomeBridge_ERC20_to_ERC20
  #initialize
     ✓ sets variables (260ms)
     ✓ cant set maxPerTx > dailyLimit (97ms)
     ✓ can be deployed via upgradeToAndCall (162ms)
     ✓ cant initialize with invalid arguments (291ms)
     ✓ can initialize with zero gas price  (66ms)
  #fallback
     ✓ reverts
  #setting limits
     ✓ #setMaxPerTx allows to set only to owner and cannot be more than daily limit (93ms)
     ✓ #setMinPerTx allows to set only to owner and cannot be more than daily limit and should be less than
maxPerTx (87ms)
  #executeAffirmation
     ✓ should allow validator to withdraw (163ms)
     ✓ should allow validator to withdraw with zero value (154ms)
     ✓ test with 2 signatures required (469ms)
     ✓ should not allow to double submit (87ms)
     ✓ should not allow non-authorities to execute deposit
     ✓ doesnt allow to deposit if requiredSignatures has changed (505ms)
     ✓ works with 5 validators and 3 required signatures (361ms)
     ✓ should not allow execute affirmation over foreign max tx limit
     ✓ should fail if txHash already set as above of limits (120ms)
     ✓ should not allow execute affirmation over daily foreign limit (291ms)
```

```
#isAlreadyProcessed
    ✓ returns  (89ms)
#submitSignature
    ✓ allows a validator to submit a signature (121ms)
    ✓ when enough requiredSignatures are collected, CollectedSignatures event is emitted (244ms)
    ✓ works with 5 validators and 3 required signatures (410ms)
    ✓ attack when increasing requiredSignatures (322ms)
    ✓ attack when decreasing requiredSignatures (162ms)
#requiredMessageLength
    ✓ should return the required message length
#fixAssetsAboveLimits
    ✓ Should revert if value to unlock is bigger than max per transaction (88ms)
    ✓ Should allow to partially reduce outOfLimitAmount and not emit UserRequestForSignature (162ms)
    ✓ Should allow to partially reduce outOfLimitAmount and emit UserRequestForSignature (158ms)
    ✓ Should revert if try to unlock more than available (249ms)
    ✓ Should not be allow to be called by an already fixed txHash (326ms)
    ✓ Should fail if txHash didnt increase out of limit amount (82ms)
    ✓ Should fail if not called by proxyOwner (113ms)
    ✓ Should emit UserRequestForSignature with value reduced by fee (195ms)
#claimTokens
    ✓ should be able to call claimTokens on tokenAddress (512ms)
#rewardableInitialize
    ✓ sets variables (459ms)
    ✓ can update fee contract (161ms)
    ✓ can update fee (182ms)
    ✓ fee should be less than 100% (316ms)
    ✓ should be able to get fee manager mode (108ms)
    ✓ should be able to set blockReward contract (305ms)
  #onTokenTransfer
    ✓ should trigger UserRequestForSignature with transfer value (197ms)
    ✓ should be able to specify a different receiver (246ms)
    ✓ should trigger UserRequestForSignature with fee subtracted (435ms)
  #rewardable_submitSignatures
    ✓ should distribute fee to one validator (287ms)
    ✓ should distribute fee to 3 validators (365ms)
    ✓ should distribute fee to 5 validators (456ms)
  #rewardable_executeAffirmation
    ✓ should distribute fee to one validator (277ms)
    ✓ should distribute fee to 3 validators (347ms)
    ✓ should distribute fee to 5 validators (473ms)
  #decimals Shift
    ✓ Foreign to Home: works with 5 validators and 3 required signatures with decimal shift 2 (387ms)
    ✓ Foreign to Home: test decimal shift 2, no impact on UserRequestForSignature value (303ms)

Contract: ForeignBridge_ERC20_to_Native
  #initialize
    ✓ should initialize (647ms)
  #executeSignatures
    ✓ should allow to executeSignatures (123ms)
    ✓ should allow second withdrawal with different transactionHash but same recipient and value (218ms)
    ✓ should not allow second withdraw (replay attack) with same transactionHash but different recipient
(168ms)
    ✓ should not allow withdraw over home max tx limit (86ms)
    ✓ should not allow withdraw over daily home limit (197ms)
  #withdraw with 2 minimum signatures
    ✓ withdraw should fail if not enough signatures are provided (123ms)
    ✓ withdraw should fail if duplicate signature is provided (81ms)
    ✓ works with 5 validators and 3 required signatures (318ms)
  #upgradeable
    ✓ can be upgraded (436ms)
    ✓ can be deployed via upgradeToAndCall (155ms)
  #claimTokens
    ✓ can send erc20 (433ms)
  #decimalShift
    ✓ Home to Foreign: withdraw with 1 signature with a decimalShift of 2 (464ms)
    ✓ Home to Foreign: withdraw with 2 minimum signatures with a decimalShift of 2 (348ms)
  #relayTokens
    ✓ should allow to bridge tokens using approve and relayTokens (388ms)
    ✓ should allow to bridge tokens using approve and relayTokens with different recipient (241ms)
    ✓ should allow only sender to specify a different receiver (312ms)
    ✓ should not be able to transfer more than limit (277ms)
    ✓ should allow to call relayTokens without specifying the sender (200ms)
  migrateToMCD
    ✓ should be able to swap tokens (269ms)
  support two tokens
    isTokenSwapAllowed
      ✓ isTokenSwapAllowed should return true if SCD ES was executed (111ms)
    isHDTokenBalanceAboveMinBalance
      ✓ isHDTokenBalanceAboveMinBalance should return true if balance above the threshold  (109ms)
    halfDuplexErc20token
      ✓ should be able to get half duplex erc20 token
    swapTokens
      ✓ should be able to swap tokens calling swapTokens (482ms)
    relayTokens
      ✓ should allow to bridge tokens specifying the token address (290ms)
      ✓ should use erc20Token if token address is zero (134ms)
      ✓ should swap token if half duplex token is used (200ms)
      ✓ should fail if token address is unknown (126ms)
      ✓ should allow specify the sender and a different receiver (348ms)
      ✓ should not be able to transfer more than limit (240ms)
    onExecuteMessage
      ✓ should swapTokens in executeSignatures (226ms)
    claimTokens
      ✓ can send erc20 (414ms)

Contract: HomeBridge_ERC20_to_Native
  #initialize
    ✓ sets variables (290ms)
    ✓ can update block reward contract (383ms)
    ✓ cant set maxPerTx > dailyLimit (95ms)
    ✓ can be deployed via upgradeToAndCall (227ms)
    ✓ can be upgraded keeping the state (348ms)
    ✓ cant initialize with invalid arguments (231ms)
```

```
#rewardableInitialize
    ✓ sets variables (432ms)
    ✓ cant initialize with invalid arguments (298ms)
    ✓ can update fee contract (126ms)
    ✓ can update fee (141ms)
    ✓ fee should be less than 100% (276ms)
#fallback
    ✓ should accept native coins (137ms)
    ✓ should accumulate burnt coins (207ms)
    ✓ doesnt let you send more than daily limit (297ms)
    ✓ doesnt let you send more than max amount per tx (257ms)
    ✓ should not let to deposit less than minPerTx (190ms)
    ✓ should fail if not enough bridged tokens (212ms)
#relayTokens
    ✓ should accept native coins and alternative receiver (131ms)
    ✓ should accumulate burnt coins (201ms)
    ✓ doesnt let you send more than daily limit (292ms)
    ✓ doesnt let you send more than max amount per tx (247ms)
    ✓ should not let to deposit less than minPerTx (201ms)
    ✓ should fail if not enough bridged tokens (217ms)
#setting limits
    ✓ setMaxPerTx allows to set only to owner and cannot be more than daily limit (105ms)
    ✓ setMinPerTx allows to set only to owner and cannot be more than daily limit and should be less than
maxPerTx (102ms)
      ✓ setExecutionMaxPerTx allows to set only to owner and cannot be more than execution daily limit (113ms)
      ✓ executionDailyLimit allows to set only to owner (198ms)
  #executeAffirmation
    ✓ should allow validator to executeAffirmation (69ms)
    ✓ should allow validator to executeAffirmation with zero value (59ms)
    ✓ test with 2 signatures required (359ms)
    ✓ should not allow non-validator to execute affirmation
    ✓ should fail if the block reward contract is not set (282ms)
    ✓ works with 5 validators and 3 required signatures (296ms)
    ✓ should not allow execute affirmation over foreign max tx limit (38ms)
    ✓ should fail if txHash already set as above of limits (111ms)
    ✓ should not allow execute affirmation over daily foreign limit (229ms)
  #submitSignature
    ✓ allows a validator to submit a signature (107ms)
    ✓ when enough requiredSignatures are collected, CollectedSignatures event is emitted (216ms)
    ✓ works with 5 validators and 3 required signatures (335ms)
    ✓ attack when increasing requiredSignatures (763ms)
    ✓ attack when decreasing requiredSignatures (173ms)
  #requiredMessageLength
    ✓ should return the required message length
  #fixAssetsAboveLimits
    ✓ Should revert if value to unlock is bigger than max per transaction (87ms)
    ✓ Should allow to partially reduce outOfLimitAmount and not emit UserRequestForSignature (149ms)
    ✓ Should allow to partially reduce outOfLimitAmount and emit UserRequestForSignature (161ms)
    ✓ Should revert if try to unlock more than available (237ms)
    ✓ Should not be allow to be called by an already fixed txHash (387ms)
    ✓ Should fail if txHash didnt increase out of limit amount (80ms)
    ✓ Should fail if not called by proxyOwner (110ms)
    ✓ Should emit UserRequestForSignature with value reduced by fee (444ms)
#feeManager
    ✓ should be able to set and get fee manager contract (78ms)
    ✓ should be able to set and get fees (159ms)
    ✓ should be able to get fee manager mode (79ms)
    ✓ should be able to get fee manager mode from POSDAO fee manager (77ms)
#feeManager_ExecuteAffirmation
    ✓ should distribute fee to validator (440ms)
    ✓ should distribute fee to 3 validators (564ms)
    ✓ should distribute fee to 5 validators (869ms)
#feeManager_fallback
    ✓ should subtract fee from value (189ms)
#feeManager_relayTokens
    ✓ should subtract fee from value (185ms)
#feeManager_submitSignature
    ✓ should distribute fee to validator (493ms)
    ✓ should distribute fee to 3 validators (634ms)
    ✓ should distribute fee to 5 validators (759ms)
#FeeManager_random
    ✓ should return value between 0 and 3 (448ms)
#feeManager_ExecuteAffirmation_POSDAO
    ✓ should distribute fee to validator (781ms)
    ✓ should distribute fee to 3 validators (595ms)
    ✓ should distribute fee to 5 validators (720ms)
#feeManager_fallback_POSDAO
    ✓ should subtract fee from value (174ms)
  #feeManager_relayTokens_POSDAO
    ✓ should subtract fee from value (178ms)
  #feeManager_submitSignature_POSDAO
    ✓ should distribute fee to validator (599ms)
    ✓ should distribute fee to 3 validators (683ms)
    ✓ should distribute fee to 5 validators (809ms)
  #decimals Shift
    ✓ Foreign to Home: test with 2 signatures required and decimal shift 2 (438ms)
    ✓ Home to Foreign: test decimal shift 2, no impact on UserRequestForSignature value (334ms)

Contract: ArbitraryMessage.sol
  unpackData
    ✓ unpack dataType 0x00 (100ms)
    ✓ unpack dataType 0x01 (45ms)
    ✓ unpack dataType 0x02 (45ms)
  unpackData with signatures parameters
    ✓ unpack dataType 0x00 (45ms)
    ✓ unpack dataType 0x01 (49ms)
    ✓ unpack dataType 0x02 (45ms)

Contract: ForeignBridge
  #initialize
    ✓ should initialize (569ms)
  #executeSignatures
    ✓ should allow to deposit (128ms)
```

```
      ✓ should reject if address is not foreign address (68ms)
      ✓ should allow second deposit with different transactionHash but same recipient and value (199ms)
      ✓ should not allow second deposit (replay attack) with same transactionHash but different recipient
(138ms)
      ✓ should not allow withdraw over home max tx limit (55ms)
      ✓ should not allow withdraw over daily home limit (155ms)
    #executeSignatures with 2 minimum signatures
      ✓ deposit should fail if not enough signatures are provided (128ms)
      ✓ deposit should fail if duplicate signature is provided (80ms)
      ✓ works with 5 validators and 3 required signatures (317ms)
      ✓ Should fail if length of signatures is not equal (356ms)
    #onTokenTransfer
      ✓ can only be called from token contract (274ms)
      ✓ should not allow to burn more than the limit (438ms)
      ✓ should only let to send within maxPerTx limit (420ms)
      ✓ should not let to withdraw less than minPerTx (315ms)
      ✓ should be able to specify a different receiver (381ms)
    #setting limits
      ✓ #setMaxPerTx allows to set only to owner and cannot be more than daily limit (83ms)
      ✓ #setMinPerTx allows to set only to owner and cannot be more than daily limit and should be less than
maxPerTx (92ms)
    #upgradeable
      ✓ can be upgraded (482ms)
      ✓ can be deployed via upgradeToAndCall (194ms)
      ✓ can transfer ownership (183ms)
    #claimTokens
      ✓ can send erc20 (427ms)
      ✓ also calls claimTokens on tokenAddress (419ms)
      ✓ works with token that not return on transfer (384ms)
    #rewardableInitialize
      ✓ sets variables (541ms)
      ✓ can update fee contract (162ms)
      ✓ can update fee (120ms)
      ✓ fee should be less than 100% (179ms)
      ✓ should be able to get fee manager mode (96ms)
    #RewardableBridge_executeSignatures
      ✓ should distribute fee to validator (253ms)
      ✓ should distribute fee to 3 validators (395ms)
      ✓ should distribute fee to 5 validators (441ms)
    #decimalShift
      ✓ Home to Foreign: withdraw works with decimalShift of 2 (351ms)
      ✓ Foreign to Home: no impact in transferAndCall event signal for bridges oracles with a decimalShift of 2.
(239ms)

  Contract: HomeBridge
    #initialize
      ✓ sets variables (249ms)
      ✓ cant set maxPerTx > dailyLimit (94ms)
      ✓ can set gas Price  (147ms)
      ✓ can set Required Block Confirmations (151ms)
      ✓ can be deployed via upgradeToAndCall (242ms)
      ✓ cant initialize with invalid arguments (157ms)
      ✓ can transfer ownership (155ms)
      ✓ can transfer proxyOwnership (115ms)
    #fallback
      ✓ should accept native coins (179ms)
      ✓ doesnt let you send more than max amount per tx (248ms)
      ✓ should not let to deposit less than minPerTx (135ms)
    #relayTokens
      ✓ should accept native coins and alternative receiver (171ms)
      ✓ doesnt let you send more than max amount per tx (207ms)
      ✓ should not let to deposit less than minPerTx (150ms)
    #setting limits
      ✓ #setMaxPerTx allows to set only to owner and cannot be more than daily limit (94ms)
      ✓ #setMinPerTx allows to set only to owner and cannot be more than daily limit and should be less than
maxPerTx (84ms)
      ✓ #setDailyLimit allow to set by owner and should be greater than maxPerTx or zero (187ms)
    #executeAffirmation
      ✓ should allow validator to executeAffirmation (63ms)
      ✓ should allow validator to executeAffirmation with zero value (55ms)
      ✓ test with 2 signatures required (377ms)
      ✓ should not allow to double submit (81ms)
      ✓ should not allow non-authorities to execute withdraw
      ✓ doesnt allow to withdraw if requiredSignatures has changed (413ms)
      ✓ force withdraw if the recepient has fallback to revert (106ms)
      ✓ works with 5 validators and 3 required signatures (312ms)
      ✓ should not allow execute affirmation over foreign max tx limit (40ms)
      ✓ should not allow execute affirmation over daily foreign limit (183ms)
    #isAlreadyProcessed
      ✓ returns  (80ms)
    #submitSignature
      ✓ allows a validator to submit a signature (106ms)
      ✓ when enough requiredSignatures are collected, CollectedSignatures event is emitted (215ms)
      ✓ works with 5 validators and 3 required signatures (368ms)
      ✓ attack when increasing requiredSignatures (329ms)
      ✓ attack when decreasing requiredSignatures (160ms)
    #requiredMessageLength
      ✓ should return the required message length
    #claimTokens
      ✓ should work with token that return bool on transfer (342ms)
      ✓ should works with token that not return on transfer (346ms)
      ✓ should work for native coins (158ms)
    #rewardableInitialize
      ✓ sets variables (435ms)
      ✓ can update fee contract (158ms)
      ✓ can update fee (122ms)
      ✓ fee should be less than 100% (210ms)
      ✓ should be able to get fee manager mode (93ms)
      ✓ should be able to get fee manager mode for both directions (130ms)
    #feeManager_OneDirection_fallback
      ✓ should not subtract fee from value (221ms)
    #feeManager_OneDirection_relayRequest
      ✓ should not subtract fee from value (225ms)
```

```
  #feeManager_OneDirection_submitSignature
     ✓ should not distribute fee to validator (285ms)
  #feeManager_OneDirection_ExecuteAffirmation
     ✓ should distribute fee to validator (294ms)
     ✓ should distribute fee to 3 validators (488ms)
     ✓ should distribute fee to 5 validators (501ms)
  #feeManager_BothDirections_fallback
     ✓ should subtract fee from value (227ms)
  #feeManager_BothDirections_relayRequest
     ✓ should subtract fee from value (226ms)
  #feeManager_BothDirections_submitSignature
     ✓ should distribute fee to validator (314ms)
     ✓ should distribute fee to 3 validators (463ms)
     ✓ should distribute fee to 5 validators (606ms)
  #feeManager_BothDirections_ExecuteAffirmation
     ✓ should distribute fee to validator (291ms)
     ✓ should distribute fee to 3 validators (364ms)
     ✓ should distribute fee to 5 validators (527ms)
  #decimalShift
     ✓ Foreign to Home: works with 5 validators and 3 required signatures with decimal shift 2 (324ms)
     ✓ Foreign to Home: test decimal shift 2, no impact on UserRequestForSignature value (249ms)

Contract: ERC677BridgeToken
   ✓ default values (89ms)
#bridgeContract
   ✓ can set bridge contract (96ms)
   ✓ only owner can set bridge contract (137ms)
   ✓ fail to set invalid bridge contract address (142ms)
#mint
   ✓ can mint by owner (67ms)
   ✓ no one can call finishMinting
   ✓ cannot mint by non-owner (65ms)
#transfer
   ✓ sends tokens to recipient (113ms)
   ✓ sends tokens to bridge contract (169ms)
   ✓ sends tokens to contract that does not contains onTokenTransfer method (85ms)
   ✓ fail to send tokens to bridge contract out of limits (167ms)
#transferFrom
   ✓ should call onTokenTransfer (226ms)
#burn
   ✓ can burn (109ms)
#transferAndCall
   ✓ calls contractFallback (284ms)
   ✓ sends tokens to bridge contract (166ms)
   ✓ fail to sends tokens to contract that does not contains onTokenTransfer method (98ms)
   ✓ fail to send tokens to bridge contract out of limits (167ms)
#claimtokens
   ✓ can take send ERC20 tokens (203ms)
   ✓ works with token that not return on transfer (338ms)
#transfer
   ✓ if transfer called on contract, onTokenTransfer is also invoked (203ms)
   ✓ if transfer called on contract, still works even if onTokenTransfer doesnot exist (153ms)
  #renounceOwnership
     ✓ should not be able to renounce ownership (56ms)

Contract: ERC677BridgeTokenRewardable
   ✓ default values (123ms)
  #bridgeContract
     ✓ can set bridge contract (95ms)
     ✓ only owner can set bridge contract (141ms)
     ✓ fail to set invalid bridge contract address (98ms)
  #blockRewardContract
     ✓ can set BlockReward contract (82ms)
     ✓ only owner can set BlockReward contract (122ms)
     ✓ fail to set invalid BlockReward contract address (101ms)
  #stakingContract
     ✓ can set Staking contract (82ms)
     ✓ only owner can set Staking contract (124ms)
     ✓ fail to set invalid Staking contract address (96ms)
     ✓ fail to set Staking contract address with non-zero balance (145ms)
  #mintReward
     ✓ can only be called by BlockReward contract (79ms)
     ✓ should increase totalSupply and balance (100ms)
  #stake
     ✓ can only be called by Staking contract (141ms)
     ✓ should revert if user doesn't have enough balance (121ms)
     ✓ should decrease user's balance and increase Staking's balance (164ms)
#mint
   ✓ can mint by owner (66ms)
   ✓ no one can call finishMinting
   ✓ cannot mint by non-owner (83ms)
#transfer
   ✓ sends tokens to recipient (115ms)
   ✓ sends tokens to bridge contract (180ms)
   ✓ sends tokens to contract that does not contains onTokenTransfer method (92ms)
   ✓ fail to send tokens to bridge contract out of limits (166ms)
   ✓ fail to send tokens to BlockReward contract directly (121ms)
   ✓ fail to send tokens to Staking contract directly (112ms)
#transferFrom
   ✓ should call onTokenTransfer (222ms)
   ✓ fail to send tokens to BlockReward contract directly (147ms)
   ✓ fail to send tokens to Staking contract directly (143ms)
#burn
   ✓ can burn (115ms)
#transferAndCall
   ✓ calls contractFallback (294ms)
   ✓ sends tokens to bridge contract (167ms)
   ✓ fail to sends tokens to contract that does not contains onTokenTransfer method (98ms)
   ✓ fail to send tokens to bridge contract out of limits (186ms)
#claimtokens
   ✓ can take send ERC20 tokens (200ms)
   ✓ works with token that not return on transfer (184ms)
#transfer
```

```
      ✓ if transfer called on contract, onTokenTransfer is also invoked (203ms)
      ✓ if transfer called on contract, still works even if onTokenTransfer doesnot exist (241ms)
  #renounceOwnership
      ✓ should not be able to renounce ownership (53ms)

Contract: RewardableValidators
  #initialize
      ✓ sets values (396ms)
      ✓ should fail if exceed amount of validators (884ms)
  #addValidator
      ✓ adds validator (109ms)
      ✓ cannot add already existing validator (52ms)
      ✓ cannot add 0xf as validator address
      ✓ cannot add 0x0 as validator address
      ✓ cannot add 0x0 as reward address
  #removeValidator
      ✓ removes validator (95ms)
      ✓ cannot remove if it will break requiredSignatures (139ms)
      ✓ cannot remove non-existent validator (89ms)
  #setRequiredSignatures
      ✓ sets req signatures (79ms)
      ✓ cannot set more than  validators count (54ms)
  #upgradable
      ✓ can be upgraded via upgradeToAndCall (172ms)
  #single list remove
      ✓ should remove 0xDf08F82De32B8d460adbE8D72043E3a7e25A3B39 - without Proxy (86ms)
      ✓ Removed validator should return zero address on nextValidator (114ms)
      ✓ should remove 0xDf08F82De32B8d460adbE8D72043E3a7e25A3B39 - with Proxy (255ms)
      ✓ should remove 0x6704Fbfcd5Ef766B287262fA2281C105d57246a6 - with Proxy (219ms)
      ✓ should remove 0x9E1Ef1eC212F5DFfB41d35d9E5c14054F26c6560 - with Proxy (219ms)
      ✓ should remove 0xce42bdB34189a93c55De250E011c68FaeE374Dd3 - with Proxy (222ms)
      ✓ should remove 0x97A3FC5Ee46852C1Cf92A97B7BaD42F2622267cC - with Proxy (217ms)
  #reward address
      ✓ reward address is properly assigned (378ms)
  #Validators list
      ✓ should return validators list (75ms)

Contract: BridgeValidators
  #initialize
      ✓ sets values (326ms)
      ✓ should fail if exceed amount of validators (617ms)
  #addValidator
      ✓ adds validator (101ms)
      ✓ cannot add already existing validator (56ms)
      ✓ cannot add 0xf as validator address
      ✓ cannot add 0x0 as validator address
  #removeValidator
      ✓ removes validator (104ms)
      ✓ cannot remove if it will break requiredSignatures (138ms)
      ✓ cannot remove non-existent validator (92ms)
  #setRequiredSignatures
      ✓ sets req signatures (85ms)
      ✓ cannot set more than  validators count (52ms)
  #upgradable
      ✓ can be upgraded via upgradeToAndCall (165ms)
  #single list remove
      ✓ should remove 0xDf08F82De32B8d460adbE8D72043E3a7e25A3B39 - without Proxy (83ms)
      ✓ Removed validator should return zero address on nextValidator (108ms)
      ✓ should remove 0xDf08F82De32B8d460adbE8D72043E3a7e25A3B39 - with Proxy (202ms)
      ✓ should remove 0x6704Fbfcd5Ef766B287262fA2281C105d57246a6 - with Proxy (250ms)
      ✓ should remove 0x9E1Ef1eC212F5DFfB41d35d9E5c14054F26c6560 - with Proxy (200ms)
      ✓ should remove 0xce42bdB34189a93c55De250E011c68FaeE374Dd3 - with Proxy (211ms)
      ✓ should remove 0x97A3FC5Ee46852C1Cf92A97B7BaD42F2622267cC - with Proxy (212ms)
  #Validators list
      ✓ should return validators list (67ms)
  #isValidatorDuty
      ✓ should return if provided valdidator is on duty (102ms)


  496 passing (3m)
```

Code Coverage

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/	97.83	92.11	100	100	
ERC677BridgeToken.sol	100	88.89	100	100	
ERC677BridgeTokenRewardable.sol	95.65	95	100	100	
contracts/interfaces/	100	100	100	100	
ERC677.sol	100	100	100	100	
ERC677Receiver.sol	100	100	100	100	
IAMB.sol	100	100	100	100	
IBlockReward.sol	100	100	100	100	
IBridgeValidators.sol	100	100	100	100	
IBurnableMintableERC677Token.sol	**100**	**100**	**100**	**100**	
IRewardableValidators.sol	100	100	100	100	

IScdMcdMigration.sol	100	100	100	100	
IUpgradeabilityOwnerStorage.sol	100	100	100	100	
contracts/libraries/	98.31	69.23	100	100	
ArbitraryMessage.sol	100	100	100	100	
Bytes.sol	100	100	100	100	
Message.sol	98.18	69.23	100	100	
contracts/mocks/	85	75	80.43	85	
AMBMock.sol	100	100	100	100	
BlockReward.sol	97.3	78.57	90.91	97.3	31
Box.sol	43.75	100	50	43.75	… 36,40,41,42
DaiAdapterMock.sol	100	100	100	100	
ERC20Mock.sol	100	100	100	100	
ERC677BridgeTokenRewardableMock.sol	100	100	100	100	
ERC677ReceiverTest.sol	100	100	100	100	
FeeManagerMock.sol	50	100	25	50	15
ForeignBridgeV2.sol	25	100	50	25	11,12,13
MessageTest.sol	100	100	100	100	
NoReturnTransferTokenMock.sol	90.91	50	75	90.91	14
OldBlockReward.sol	100	100	100	100	
RevertFallback.sol	100	100	100	100	
ScdMcdMigrationMock.sol	100	100	100	100	
Staking.sol	100	100	100	100	
contracts/upgradeability/	74.07	44.44	84.62	73.33	
ClassicEternalStorageProxy.sol	0	0	0	0	… 28,29,30,32
EternalStorage.sol	100	100	100	100	
EternalStorageProxy.sol	100	100	100	100	
OwnedUpgradeabilityProxy.sol	100	50	100	100	
Proxy.sol	100	50	100	100	
UpgradeabilityOwnerStorage.sol	100	100	100	100	
UpgradeabilityProxy.sol	100	66.67	100	100	
UpgradeabilityStorage.sol	100	100	100	100	
contracts/upgradeable_contracts/	99.73	88	100	99.74	
BaseBridgeValidators.sol	100	72.73	100	100	
BaseERC677Bridge.sol	100	100	100	100	
BaseFeeManager.sol	100	100	100	100	
BaseOverdrawManagement.sol	100	100	100	100	
BasicBridge.sol	100	100	100	100	
BasicForeignBridge.sol	100	87.5	100	100	
BasicHomeBridge.sol	100	95.83	100	100	
BasicTokenBridge.sol	100	100	100	100	
BlockRewardBridge.sol	100	100	100	100	
BlockRewardFeeManager.sol	100	100	100	100	
BridgeValidators.sol	100	80	100	100	
Claimable.sol	93.33	70	100	94.12	26
ERC20Bridge.sol	**100**	**100**	**100**	**100**	
ERC677Bridge.sol	100	100	100	100	

ERC677BridgeForBurnableMintableToken.sol	100	100	100	100	
ERC677Storage.sol	100	100	100	100	
FeeTypes.sol	100	100	100	100	
Initializable.sol	100	100	100	100	
InitializableBridge.sol	100	100	100	100	
MessageRelay.sol	100	100	100	100	
OtherSideBridgeStorage.sol	100	100	100	100	
OverdrawManagement.sol	100	100	100	100	
Ownable.sol	100	100	100	100	
ReentrancyGuard.sol	100	100	100	100	
RewardableBridge.sol	100	62.5	100	100	
RewardableValidators.sol	100	84.62	100	100	
Sacrifice.sol	100	100	100	100	
Upgradeable.sol	100	100	100	100	
Validatable.sol	100	100	100	100	
ValidatorStorage.sol	100	100	100	100	
ValidatorsFeeManager.sol	100	80	100	100	
VersionableBridge.sol	100	100	100	100	
contracts/upgradeable_contracts/amb_erc677_to_erc677/	100	96.43	100	100	
BasicAMBErc677ToErc677.sol	100	96.15	100	100	
ForeignAMBErc677ToErc677.sol	100	100	100	100	
HomeAMBErc677ToErc677.sol	100	100	100	100	
contracts/upgradeable_contracts/arbitrary_message/	97.41	83.33	93.02	97.41	
BasicAMB.sol	100	87.5	100	100	
BasicForeignAMB.sol	100	50	100	100	
BasicHomeAMB.sol	100	81.25	100	100	
ForeignAMB.sol	100	100	100	100	
HomeAMB.sol	100	100	100	100	
MessageDelivery.sol	100	100	100	100	
MessageProcessor.sol	88	100	78.57	88	19,27,35
contracts/upgradeable_contracts/erc20_to_erc20/	96.75	82.5	93.75	96.77	
BasicForeignBridgeErcToErc.sol	100	75	100	100	
FeeManagerErcToErcPOSDAO.sol	100	100	100	100	
ForeignBridgeErc677ToErc677.sol	80	100	75	80	32
ForeignBridgeErcToErc.sol	100	100	100	100	
HomeBridgeErcToErc.sol	95.38	90.91	90	95.38	55,67,69
HomeBridgeErcToErcPOSDAO.sol	100	50	100	100	
RewardableHomeBridgeErcToErc.sol	100	100	100	100	
contracts/upgradeable_contracts/erc20_to_native/	99.34	82.81	100	99.35	
FeeManagerErcToNative.sol	85.71	50	100	85.71	20
FeeManagerErcToNativePOSDAO.sol	100	100	100	100	
ForeignBridgeErcToNative.sol	100	78.57	100	100	
HomeBridgeErcToNative.sol	**100**	**88.24**	**100**	**100**	
RewardableHomeBridgeErcToNative.sol	100	100	100	100	

contracts/upgradeable_contracts/native_to_erc20/	94.66	78.57	90.91	94.66	
ClassicHomeBridgeNativeToErc.sol	0	100	0	0	15,24,25
FeeManagerNativeToErc.sol	80	50	100	80	19
FeeManagerNativeToErcBothDirections.sol	80	50	100	80	21
ForeignBridgeNativeToErc.sol	100	70	100	100	
HomeBridgeNativeToErc.sol	100	87.5	100	100	
RewardableForeignBridgeNativeToErc.sol	100	100	100	100	
RewardableHomeBridgeNativeToErc.sol	50	100	50	50	11,19
All files	**96.79**	**84.12**	**95.27**	**96.93**	

Appendix

File Signatures

The following are the SHA-256 hashes of the reviewed files. A file with a different SHA-256 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different SHA-256 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review.

Contracts

```
7883e80b721899f567c0b9acf8badb0cd679efedbc92050591ef76a2929281eb ./contracts/ERC677BridgeTokenRewardable.sol

c500a6e4069d29179b2ed57777d7c06fa4e29d18ea3419517afcde0a59fc3046 ./contracts/Migrations.sol

dd3a8475ab07320b30da579fb46e632006fbe3a1419bf2cea0b0997fa4992b5b ./contracts/ERC677BridgeToken.sol

7f35b050ea4690c1977fa6896819884f38e887c4a40942c25937b60c312d423e
./contracts/upgradeability/OwnedUpgradeabilityProxy.sol

4ab093b5264a7a46ab92514abdc915ef2ccb6755d4ffe619e3c24ee0bafc3cae
./contracts/upgradeability/EternalStorageProxy.sol

eb5bba7c935f1423ed3b0def96945a1b6ae74028650c8e76f8c03a022ffd4a48
./contracts/upgradeability/ClassicEternalStorageProxy.sol

dfc565cfc75b6b361144320dc804c9558ae2d1350268648db4ec0d78b0cdc0ae ./contracts/upgradeability/Proxy.sol

09ab1dbf1ce73828e6466510e5b59a369dc985c2e4bd253cf1e405daabc5ee94
./contracts/upgradeability/UpgradeabilityOwnerStorage.sol

47f69135fe82286475ba5c53829e3935e0b1ebfe6f8cbbe5114edb2523b414aa ./contracts/upgradeability/EternalStorage.sol

d779a190cc20fdd9d88c445162e914c140f1de378b77d18e00a00a0be74775bf
./contracts/upgradeability/UpgradeabilityProxy.sol

eeb90c347feb4ec45a9366af1c97ddbff79d08b74957270fdc040cbc12c22584
./contracts/upgradeability/UpgradeabilityStorage.sol

57840e5d1b6afe199b117b4ce701fa1432d2e8ba5f850dd120c6bb055375a280 ./contracts/interfaces/IAMB.sol

5f715cc665ecc40619a69d12dfdce6ba78cbd8a429ff216b87297f19a020ff12 ./contracts/interfaces/ERC677Receiver.sol

5c995c4fb0e86d94142ecaadcf683e74e6d011d95baa2944b98dba536f233d61
./contracts/interfaces/IRewardableValidators.sol

50f6f6a31cbb50264095919656b03ad641e491769950675014a7e1fce4803fd1 ./contracts/interfaces/IBlockReward.sol

817d0221c83f2b2184c71155c1577b495ec6a8e251b871aacd265706534d1f84
./contracts/interfaces/IBurnableMintableERC677Token.sol

324b4fa17b1caa6817aa9c37173f3bd8c12ce3cd6781a8438afa7930e5bfec63
./contracts/interfaces/IUpgradeabilityOwnerStorage.sol

14fc66e9789e5b58dfa55736436b0991cb7a9f2d4a5f5d175c7ab4bc7255a284 ./contracts/interfaces/ERC677.sol

44581b411f62c31e35d46e8d3ef927ddd09fd6d37744a92fcd18da63a764039f ./contracts/interfaces/IScdMcdMigration.sol

fcd0f053ac73391f42f8e46c862bb121eae9795f8e15f7ffc7047c9c720cbd6a ./contracts/interfaces/IBridgeValidators.sol

c58f3c06ce573689ae4aa435e1de18c33290ba8c47144523c3e72bf2907af881 ./contracts/libraries/Bytes.sol

bd98b6c614981c0d67efc8c2badae2c79a11ddd026ba9636c353570aff4eacf0 ./contracts/libraries/Message.sol

6da1605968f20f06fd89881c1f1b289b5fc1276d9efb892784130f04ec0fe802 ./contracts/libraries/ArbitraryMessage.sol

973a28a487bbb142ff0a35c06ae26181532911350073833252 3af69ce8719471
./contracts/upgradeable_contracts/Initializable.sol

57a299adcf5c403d445646a3a75eed4fc81870c3e5a69bb6820839b437458929 ./contracts/upgradeable_contracts/Claimable.sol

95110ed77b2b0c2847cfd84a48f72d0e984683302c8cee7ccda87b6f678168b2
./contracts/upgradeable_contracts/BlockRewardFeeManager.sol
```

```
c64dc1562dd6214485e098597371926f0deddae2ae00bccc24d69e8472f1c85d
./contracts/upgradeable_contracts/BaseERC677Bridge.sol

3376a33dd88f7fa06319717e03f284817a554968b89be27796e9dcdacacd4dbf
./contracts/upgradeable_contracts/ReentrancyGuard.sol

e5c915dbc8c92a2bb69732d8a1e2a853e17e40b85bbf1f22b45af83ad7bda458
./contracts/upgradeable_contracts/Upgradeable.sol

7b3d62ec7ec5ed1ecebe83a0c9b2155d8bca52a1bd7e420bea51577e85836f7a
./contracts/upgradeable_contracts/ValidatorStorage.sol

883caaefc73b2d1b13bfc8840d86e0b4fdab5877228352378de658d5cec0250b
./contracts/upgradeable_contracts/BasicBridge.sol

1685d92cb13ee110c2b0e229f5df3933655becf652e2a3a5e9671fe7e25f1cde
./contracts/upgradeable_contracts/VersionableBridge.sol

e1a08550b05dcfb070318bc62e98ce0bf1b4b6bccf9ac24189a66baeac5c6d9f
./contracts/upgradeable_contracts/ERC677Bridge.sol

5247f2d0100d741ac1009656647cacf52fe04e971cf34c6ea792118eb83769e0
./contracts/upgradeable_contracts/ERC677Storage.sol

286a2966e5bd4377f5f5130afb2dc1602cefb388c9b7f4690a8c717abe99bfb8 ./contracts/upgradeable_contracts/Sacrifice.sol

a936ffc60eeb5ebdde2a272ccc8bfdbe6f0f5169732db9843fd76470abfe112c
./contracts/upgradeable_contracts/BaseOverdrawManagement.sol

4ba079ae84858700e417eea5f8638708aa502eab86bd2e56f91b910ab7d76616 ./contracts/upgradeable_contracts/Ownable.sol

1121c87b80b43ffa345fa609bc407b9d920a21d455859c741e2967471d629af9
./contracts/upgradeable_contracts/InitializableBridge.sol

b9b46c8c3328ab94e0b7d8393d7193f9cedd8ac4d766a406e44ac26537066e0b
./contracts/upgradeable_contracts/RewardableValidators.sol

86806157bf77e1676869850520c4f4deb84ba0f80c7d66dc80b37be82e39ce19
./contracts/upgradeable_contracts/BridgeValidators.sol

53b862c390b9f6802a04dd52658448f0bd524ac6011cf07f382b92993d2c28cc
./contracts/upgradeable_contracts/ValidatorsFeeManager.sol

9cab9b36565d3cc0824e2c0a926dc8214033ac0594ab5a278cf685ff2c4f2c22
./contracts/upgradeable_contracts/BasicTokenBridge.sol

b5291f53c0d5438130dd81e007ffc9126d3101d75570490779ebd532c3337da3
./contracts/upgradeable_contracts/BaseBridgeValidators.sol

155b308fcc0d6d5f12ce0fd85e26b19736c34e29cce670fe619f07fb6bdab1ad
./contracts/upgradeable_contracts/Validatable.sol

ba378ea945581f4a56dec023900d33111e7a72c2ead943ee87092072abad0990
./contracts/upgradeable_contracts/BasicForeignBridge.sol

4a62bad21b3825f26f6e956e7de645728a5c9e9737da030fd2788fe34c5ea71f
./contracts/upgradeable_contracts/OverdrawManagement.sol

caa7d8497bb13267f07f307a55c824f39322a7a1589c9ea706aed35c527d88e2
./contracts/upgradeable_contracts/ERC20Bridge.sol

54963c9f7d6a8f4377e34aa4e8dcff96acad0e33bc95989535ac3f1f7df46948
./contracts/upgradeable_contracts/RewardableBridge.sol

70411ae02cbb01ff7a95fd77ae9b53cef45981a6226255882f157be4a8a0e15f
./contracts/upgradeable_contracts/ERC677BridgeForBurnableMintableToken.sol

f0d033b815c7165d18674b6a020d93ee4b87878ab52c8ae42c6073b7dc2f4f53
./contracts/upgradeable_contracts/BasicHomeBridge.sol

7abe690fb824a9fda8dd7b12efc9cbac0ed64cc6c0ca56f22577c0b80ba14e8c
./contracts/upgradeable_contracts/OtherSideBridgeStorage.sol

2265f6d1b349455e11ceae1edc029d6303d0a769d18cd2110f34f769408e3d20
./contracts/upgradeable_contracts/BlockRewardBridge.sol

576ec281e15c6e8e9c24801c81e68eb48ed0fffc7e1c725ee03c9c731b7481ec
./contracts/upgradeable_contracts/MessageRelay.sol

0df1671c6c07a2e3eb107d68aa71b98f41a9dd3716db9a0555abb1b74ff3f140
./contracts/upgradeable_contracts/BaseFeeManager.sol

adea3fdef59299bcb3da4dc750f7eb5de99179f02faba416426c9f0437abb70a ./contracts/upgradeable_contracts/FeeTypes.sol

a18bbd7c4177f6f28fd7d64e7f29d096cf7ccc2173be6bae45d95fd2a710cc9f
./contracts/upgradeable_contracts/erc20_to_native/FeeManagerErcToNativePOSDAO.sol

6d0deea6fe807545ca23e5249ce6389015dd11fe639ce86ed5635dda59de10a1
./contracts/upgradeable_contracts/erc20_to_native/ForeignBridgeErcToNative.sol

c6f274e636c8b19569b4e19591621bd616a7c10136253a69077bbceeb484c081
./contracts/upgradeable_contracts/erc20_to_native/RewardableHomeBridgeErcToNative.sol

b104160cb0934c5d8b12ce692868a1ad8003b80650a1d3c1752e2fd469d2f118
./contracts/upgradeable_contracts/erc20_to_native/HomeBridgeErcToNative.sol

d840071df5ffadb8a25b2dc222385a90a1a418f1155d43c1b36467abadb08213
./contracts/upgradeable_contracts/erc20_to_native/FeeManagerErcToNative.sol
```

```
11f5ff07c7467a778f50acc0a0e58aed146df775b968be8e6e5c7b3315780ef5
./contracts/upgradeable_contracts/erc20_to_erc20/HomeBridgeErcToErc.sol

2e0ba57492e613d5e5bc7ae1959f5293087ce079cc98349da5addafaa78efaba
./contracts/upgradeable_contracts/erc20_to_erc20/ForeignBridgeErc677ToErc677.sol

4d18ac818e1d270eefa66e55cddd72f0bfa4024ac66c8767f63752bbd4b1d06e
./contracts/upgradeable_contracts/erc20_to_erc20/BasicForeignBridgeErcToErc.sol

b66cfc63f26f6603c34dbbeccb77eca30fe7456073cbb74a74404f5070cddbb9
./contracts/upgradeable_contracts/erc20_to_erc20/HomeBridgeErcToErcPOSDAO.sol

2814531010239d5a1ae538d9c7ab50cec37283f60e6c166ab0ec4015ecc97209
./contracts/upgradeable_contracts/erc20_to_erc20/ForeignBridgeErcToErc.sol

683c374e56fabe06970efe005de8f16563cecab555cfa18593eecd2c3aa4ce17
./contracts/upgradeable_contracts/erc20_to_erc20/FeeManagerErcToErcPOSDAO.sol

b2c9f352a00b0bbdf3ca0025722d84cf569cd80adb25e5b7bdd654ab5349a13e
./contracts/upgradeable_contracts/erc20_to_erc20/RewardableHomeBridgeErcToErc.sol

fc5771360a9dcebbd69abe2e2256821fb97c229a9b5e49aa1a7f83b1a3c26fce
./contracts/upgradeable_contracts/native_to_erc20/ForeignBridgeNativeToErc.sol

f14b169021d0f544ce1c8ed8412548df4df98c0012172813e42f9399cee44ce4
./contracts/upgradeable_contracts/native_to_erc20/RewardableHomeBridgeNativeToErc.sol

609aa053f8881c5fe9dc37c28613d4a16308e4472e0687c3292543f1381c5958
./contracts/upgradeable_contracts/native_to_erc20/FeeManagerNativeToErcBothDirections.sol

edb96a8caa3998ed6ab9a99878fbadde8467921089a02f741828098afd667abd
./contracts/upgradeable_contracts/native_to_erc20/ClassicHomeBridgeNativeToErc.sol

6b09e559a25b0ca2025887de92bbceccf7ec05c41779c976e3f174ad24d76edd
./contracts/upgradeable_contracts/native_to_erc20/FeeManagerNativeToErc.sol

772e1bf59f4ed172808f5237786558c1f618c399d2a599bf4c0c3e6b4140923e
./contracts/upgradeable_contracts/native_to_erc20/HomeBridgeNativeToErc.sol

c3f0f1f9d038040b83300890d4c6ae6ba4c1e3773e996aa0f92756fbc4754598
./contracts/upgradeable_contracts/native_to_erc20/RewardableForeignBridgeNativeToErc.sol

e4e3651b577b085ef3439ff703966164dd6a396522a5e7ca24bd5fe795e66999
./contracts/upgradeable_contracts/arbitrary_message/MessageDelivery.sol

832be9796d7f26e3e57906f480e85a45c4c48a2d926fad3373bbe75c6422b06c
./contracts/upgradeable_contracts/arbitrary_message/BasicHomeAMB.sol

5a1b4cc776e11b00f4d49553ad0926ab4bf564c67d11d8de5a263dfd10da075d
./contracts/upgradeable_contracts/arbitrary_message/BasicForeignAMB.sol

40b5f36f7eac5a50ff59806f596255c89b7a3a3e3efc1eb1ef1fcc10e351d0a9
./contracts/upgradeable_contracts/arbitrary_message/HomeAMB.sol

a7fee7cce566a12a2a9afe1a810cdb946ca0b025342f10aa1931269ea48f6bf6
./contracts/upgradeable_contracts/arbitrary_message/BasicAMB.sol

7fc57349b720cd865fade20548694b7eeecae5f4ccef6ece55f4d8dacf1b3a93
./contracts/upgradeable_contracts/arbitrary_message/MessageProcessor.sol

a5860078596a755578f06181d2e91549f8a0f2c154ad4c9c414fad7ded1f5847
./contracts/upgradeable_contracts/arbitrary_message/ForeignAMB.sol

2d8deebe05d0fee0e2e75c71805ca4b5d15310c4305d4dba1806e43acccfe1a5
./contracts/upgradeable_contracts/amb_erc677_to_erc677/HomeAMBErc677ToErc677.sol

5b8c5f6d2cdcce2442e1a8b16d5a4ac7703bc3911f1cb18dbbb51e8d67a795d8
./contracts/upgradeable_contracts/amb_erc677_to_erc677/BasicAMBErc677ToErc677.sol

7e0ba7a14c7d9e508960d9a07afe8a819282648793d07960032627bbeef95b43
./contracts/upgradeable_contracts/amb_erc677_to_erc677/ForeignAMBErc677ToErc677.sol

3c1f9fe0c87839a696bf4da6d64a31699700faf3b9239874b1da87579f1bd643 ./contracts/mocks/OldBlockReward.sol

16f76f41c473ad2e9eb85814810984f6157305b9863d3a8050eb4081f9f265c9 ./contracts/mocks/BlockReward.sol

31202049d507d54df1997747f3b3b5dee22a322295d282010dffb456d1a860bd ./contracts/mocks/RevertFallback.sol

4d1cd242e8c8a301f17c9e2ab0302e5d2805d0d3ff09421a33ff38df7c4c0d1a ./contracts/mocks/AMBMock.sol

fb9673b9068c0585b43fbe067fb1c5e50ad2c734ecbfc0b8c6eb07087f2a6710 ./contracts/mocks/MessageTest.sol

63605e00156fc6385d2df9c02436a8c6464c768022f31ff102ff5a768dbf3c0a ./contracts/mocks/ScdMcdMigrationMock.sol

3618034028f3e4402b22630347781b2cb3d96f5c419117701fe33bdb34daf056 ./contracts/mocks/ERC20Mock.sol

1d239dad1a0c100eb609e4cac142df330381a022c3bac766f43cfb5630bf816a
./contracts/mocks/ERC677BridgeTokenRewardableMock.sol

8935edc88a3131e700646a9b8ed2e0449b4be8f74f5da929f82ef2d5cdeed264 ./contracts/mocks/SaiTopMock.sol

149e1384536b12d24c3c518bd928e0203339cd46449ea398e5f92450feb947a4 ./contracts/mocks/ForeignBridgeV2.sol

6c70b40317cf6ae724b33ac23535cb0434839a6f9eee079343d435c97d832752 ./contracts/mocks/NoReturnTransferTokenMock.sol

26e5ce1c5f77a98f7a0836b707b4d7cb14b8e2d9768d0c7cd815bdff53c957c1 ./contracts/mocks/Box.sol

d4620bd444f71fc318e13ee6223cbeda1741cc3f73ea2b1d61f13e4e709904fb
./contracts/mocks/ForeignBridgeErcToNativeMock.sol
```

afa64355d703716429604a0dfe7bc3b7d5e2ee3576e01f3094334972b713af6f ./contracts/mocks/ERC677ReceiverTest.sol

0ae0a3796b2ef07f20d416cbc6ac3291a29fc476d3696179530337dc87b44a0b ./contracts/mocks/DaiAdapterMock.sol

bd1517335ee79f8cf28f37f04fcfa1fe8f428d04bb284c2cfd67290f65df2b01 ./contracts/mocks/FeeManagerMock.sol

a89b26867d994d9a61b1194e89b523fc1db72d7ece57b124c5e53e8d939e1cf1 ./contracts/mocks/Staking.sol

Tests

3460ebb1ada8cb19bb483ea1991e10c0c7e4d065d01de18ff21d30d640616ae2 ./test/validators_test.js

6d91b8341cbc1b4ea005473c74bb4e7c3d2af99c42c1efe9820c0f394b428dde ./test/rewardable_validators_test.js

41ed6b7bb84e61ae0bbd590327c4f5d3187ac0b3d97400561f09859c6a8ad7d6 ./test/poa20_test.js

545e86ddae221210212213d85968817a3f94d4d0ecde47e15fe4b09549dc392b ./test/setup.js

397903dec11741aca5ad04f45d628fc04b64fbdfbf32065fcd5a5637851a6949 ./test/coverage.test.js

194a2084ec7c9bd7a986677594cc958f86d1d42cff1324e5fbea04d0515246b2 ./test/helpers/helpers.js

6dd3f3eb6da9b28f51a7af85c404cbd233c5ff66e1a4b3d2502b7fd3328d6ce6 ./test/erc_to_native/home_bridge.test.js

1346180b48836d17bd864e601df9090e2c9f44271a4b35c2cd6ef056144358e2 ./test/erc_to_native/foreign_bridge.test.js

1808a640019d28cad0c5ff3ea26e95685cd92ae8edc17cfcb258e3358b9aa73a ./test/native_to_erc/foreign_bridge_test.js

109ed7ff0aa6015a0aa6b4fa8fb6547d69169204a0c000751635e6a355e6eda0 ./test/native_to_erc/home_bridge_test.js

41ae0413c5acac5301af69c622aa587a6e542ae640e2bb95d99dacfc4625631d ./test/erc_to_erc/home_bridge.test.js

11a16ddf2b4c7931dc031916950e0ba1dc2b7b5f2588b8e94224111440eb0d57 ./test/erc_to_erc/foreign_bridge.test.js

2975ae711981daab3a5b34d20a0c4d72f4ec18a275537072838fb8bd840753e4 ./test/arbitrary_message/home_bridge.test.js

868e78bd8aac2e3a8e81fa861b2667e2e1ac187b9d40a86663a8ef560636ded7 ./test/arbitrary_message/foreign_bridge.test.js

ad2f3a5004e5ec82b49778dea0d00feebe098b8cb42adcb5ebe03ae608bdb9a2 ./test/libraries/arbitraryMessage.test.js

fd4dc4d1e76f0c459fc51371d475d2f5e3a00074afc443ae3f8e9c5eb7f01b20 ./test/amb_erc677_to_erc677/AMBErc677ToErc677Behavior.test.js

cd2fc1faac58f758f557fd21256d2ac6ef246b67a1434ff3c803c076e40a8a63 ./test/amb_erc677_to_erc677/home_bridge.test.js

33f1e72c3ccd06ab808a8b41615ad2b24752d892ba20a54a54cdf18a510df425 ./test/amb_erc677_to_erc677/foreign_bridge.test.js

About Quantstamp

Quantstamp is a Y Combinator-backed company that helps to secure smart contracts at scale using computer-aided reasoning tools, with a mission to help boost adoption of this exponentially growing technology.

Quantstamp's team boasts decades of combined experience in formal verification, static analysis, and software verification. Collectively, our individuals have over 500 Google scholar citations and numerous published papers. In its mission to proliferate development and adoption of blockchain applications, Quantstamp is also developing a new protocol for smart contract verification to help smart contract developers and projects worldwide to perform cost-effective smart contract security audits.

To date, Quantstamp has helped to secure hundreds of millions of dollars of transaction value in smart contracts and has assisted dozens of blockchain projects globally with its white glove security auditing services. As an evangelist of the blockchain ecosystem, Quantstamp assists core infrastructure projects and leading community initiatives such as the Ethereum Community Fund to expedite the adoption of blockchain technology.

Finally, Quantstamp's dedication to research and development in the form of collaborations with leading academic institutions such as National University of Singapore and MIT (Massachusetts Institute of Technology) reflects Quantstamp's commitment to enable world-class smart contract innovation.